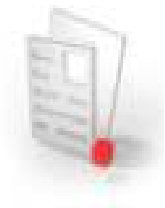




California Enterprise Architecture Program



# SOA *White Paper*

California Service Centers

## Revision History

---

08/30/2006    Original Draft

# Table of Contents

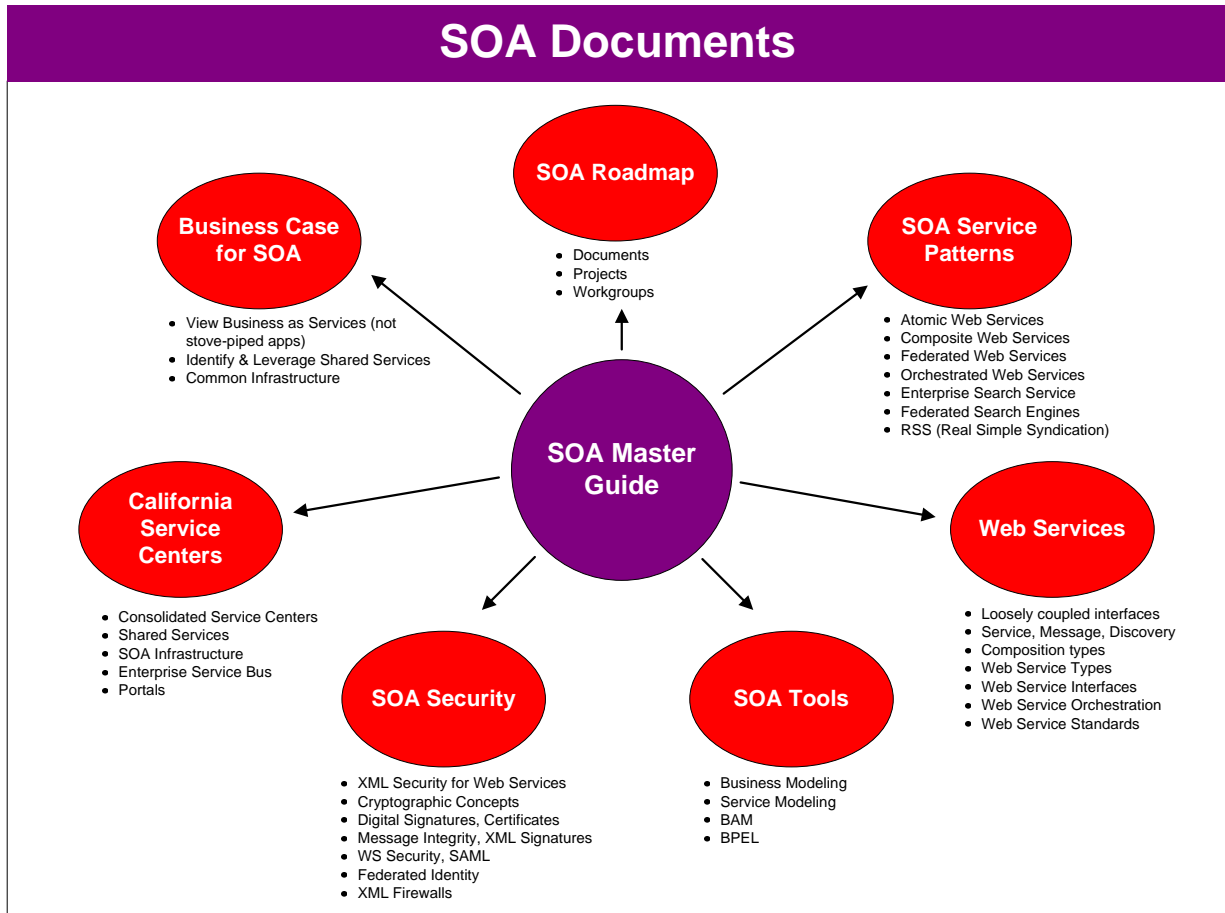
---

<b>SOA Documents .....</b>	<b>5</b>
<b>QuickView – California Service Centers .....</b>	<b>6</b>
<b>Introduction .....</b>	<b>9</b>
<b>Consolidated Service Centers .....</b>	<b>11</b>
Master and Customer-Oriented Service Centers .....	11
Federated Development .....	12
<b>Shared Services .....</b>	<b>14</b>
Simply Shared Services .....	14
Search Service .....	14
RSS Service .....	14
Identity Service .....	15
Payment Service .....	15
Employee HR and Financial Services .....	16
Other Shared Services .....	17
<b>Consolidated Infrastructure .....</b>	<b>18</b>
Enterprise Repository .....	18
Platforms .....	18
SOA Infrastructure .....	18
Labs .....	20
Test & QA Lab .....	20
Security Lab .....	20
Technical Administration .....	20
Installation .....	20
Troubleshooting .....	20
Backup & Version Rollback .....	21
Service Management .....	21
Configuration & Release Management .....	21
Service Compliance Testing .....	21
Service Performance .....	22
Common Service Desk .....	22
Service Recoverability & Disaster Management .....	22
Training .....	23
Business Activity Monitoring (BAM) .....	24
Security .....	27

**Packaged Portals ..... 28**

## SOA Documents

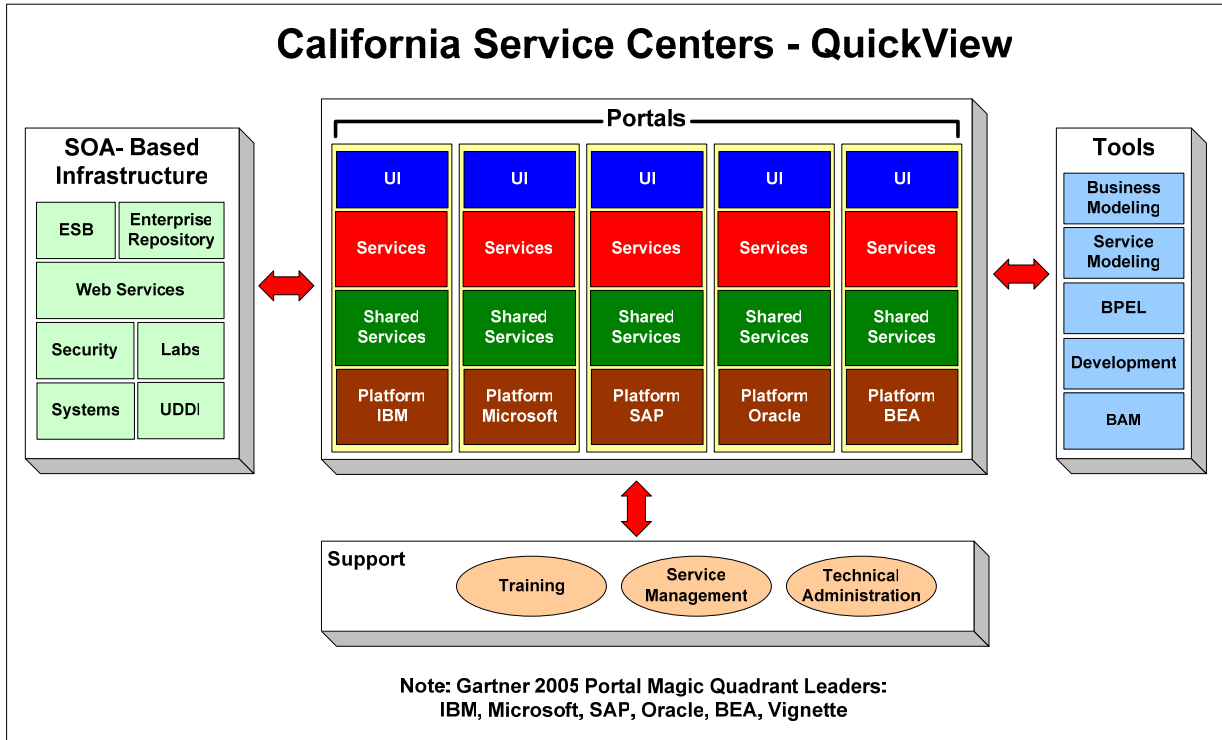
The service oriented architecture advocated by the California Enterprise Architecture Program is organized into a set of interrelated documents. A master guide serves as the “jumping off point” and describes in an overview fashion the key parts of SOA.



There are six white papers planned to address in depth details of SOA. This whitepaper is *California Service Centers*.

## QuickView – California Service Centers

The intent of this QuickView is to provide the reader with just the key points regarding California Service Centers. See the main document for more in depth details.



- Portals and Service Centers

Note portals are a packaged product provided by a vendor. Service centers are implemented via a portal. *Each* service center must be based on a *single* portal (that is, don't mix and match IBM and Oracle portals in the *same* service center since this will increase cost and complexity). Service centers can each be based on different portals; however it is highly recommended that portal platforms be limited to Gartner's upper right-hand quadrant (listed in the note on the above diagram).

A portal normally includes key portal components like configurable GUI, content management, a repository for managing its configurations and user preferences, portal development and deployment tools. Some portals also include business modeling and simulation tools. If a given service center is based on a single portal platform, then it is easier and less costly to learn one set of tools for the service center. See [Consolidated Service Centers](#) for more details.

- Portals, SOA Infrastructure, and Interoperability

Since California has adopted an SOA-based infrastructure, it is important that portal platforms are designed with SOA in mind. This is important in two areas. First, a portal might consume

or provide a shared service. Second, some portal services will need to interoperate with external applications including legacy and third party systems. So, the selected portal platform must be compliant with the state's SOA-based infrastructure particularly the ESB, Web Services, and Security components. See [Consolidated Infrastructure](#) for more details.

- Common, User-Friendly UI

One of the elements described in [Clark Kelso's "California In-Touch"](#) vision document is to replace the many state web sites with a "master service center" which would be the entry point for all users, and a relatively small number of federated "customer-oriented service centers." The idea is the master service center would have well thought out user interface which would make it easy for users to quickly find what they are looking for then be redirected to the service center designed to handle a particular type of business interaction.

The user experience would be dramatically enhanced if the common components of the user interface were standardized across service centers. That is, menus, navigation sections, and common action buttons were in the same location, labeled the same, and behaved the same across web applications. Common branding (main images and stylesheets) help ensure that California looks like a professional business instead of a dysfunctional family of service centers.

The State's IOUCA workgroup has set standards for *usability*

[http://www.cio.ca.gov/PDFs/IOUCA/IOUCA\\_Usability\\_Recommendation\\_Adopted\\_071406.pdf](http://www.cio.ca.gov/PDFs/IOUCA/IOUCA_Usability_Recommendation_Adopted_071406.pdf) and *accessibility*

[http://www.cio.ca.gov/PDFs/IOUCA/IOUCA\\_Accessibility\\_Recommendation\\_Adopted\\_071406.pdf](http://www.cio.ca.gov/PDFs/IOUCA/IOUCA_Accessibility_Recommendation_Adopted_071406.pdf), and *separating content from presentation*

[http://www.cio.ca.gov/PDFs/IOUCA/IOUCA\\_Separating\\_Content\\_from\\_Presentation\\_Recommendation\\_Adopted\\_071406.pdf](http://www.cio.ca.gov/PDFs/IOUCA/IOUCA_Separating_Content_from_Presentation_Recommendation_Adopted_071406.pdf).

- Portal Services & Shared Services

A portal will provide services that are unique to that portal (customer-oriented service center). However, under the covers a given portal service may invoke one or more shared services. For example, the Business Service Center might offer an Apply for Business License service that consumes Address Verification and Payment shared services as part of its business process for issuing a new business license.

Therefore, portal platforms and tools must be consistent with the state's SOA-based infrastructure. Since shared services are based on web services, both the data center infrastructure and portal architecture must support the same web service standards. See [Shared Services](#) for more details.

- Portal Platforms

A portal platform consists of a packaged set of components from a given vendor that together, allow a business centric portal to be developed, deployed, and managed. All of the portal vendors listed in Gartner's upper right-hand quadrant have a rich set of components. However, the packages are different and therefore require skill sets unique to that vendors packaging. There are cost, training, and operational issues that must be budgeted and planned for regardless of the package chosen. Therefore, if the state limits its portals to a very small number of vendors, the overall costs and people skills will be significantly lower.

Portals must run on a common SOA-based infrastructure to leverage cost, reduce complexity, and minimize resource skill requirements. This will allow portals to interoperate and interact with other systems much easier. See [Packaged Portals](#) for more details.

- Portal Tools

As we move from an environment of many stand-alone web sites to federated service centers, business analysts must play a much stronger role. Portals will provide services, but which business services? What are the exact processes associated with each identified business service? Which services could be shared services? How will the shared service interface be defined?

The best way to answer these questions is to use a business service modeling tool. Here all of the details regarding a specific business process can be captured. Additionally, most tools in this category allow performance metrics to be entered so simulations can be run.

Once a business service has been successfully modeled, the components that will implement the business service can then be modeled from a technical standpoint. It is important that the business and technical modeling tools be compatible. That is, the output from the business modeling tool should feed directly into the technical modeling tool without transformation. Some platforms have mechanisms that can capture actual operating statistics and feed them back into the business models to make them more accurate (since the metrics originally entered were guesses).

BAM (Business Activity Monitoring) tools can provide a real-time view of how well a business process is performing. See [BAM tools](#) for more details.

- Portal Performance

Availability, scalability, and recoverability will be very important in a shared services environment. Portals that depend on key underlying shared services will not be well accepted if they are either slow or not available. Therefore, portal and SOA infrastructures must be carefully designed to ensure appropriate availability.

In a number of cases, this means that a shared service (for example, Payment Service) must be deployed into multiple data centers. This implies collaboration among data centers to be able to redirect user requests, load balance, and keep versions of the services synchronized. See [Service Performance](#) for more details.

- Portal Support

A variety of training will be required for management, developer, business analysts, and technical administrators. There should be baseline training that people in all roles would attend as well as role-specific training. See [SOA Training](#) for a list of suggested training classes as well as how the training might be conducted.

A comprehensive ITIL-based service management program will greatly minimize problems with federated service centers. This includes shared service configuration and release

management, service compliance testing, service performance testing, a common service (help) desk, and a service recovery plan. See [Service Management](#) for more details.

Portals are complex suites of products. And, there will be multiple portals from different vendors. Therefore, installation and troubleshooting skills must to be developed and honed. See [Technical Administration](#) for more details.

## Introduction

---

This whitepaper will provide architectural details for planning and implementing new service centers in California. Please read “[Government Services on the Web: California In-Touch](#)”, authored by Clark Kelso, State CIO on 4/21/2006. This document introduces the notion of providing a better customer experience at reduced cost to the state via moving to consolidated service centers.

The new California Service Center (CSC) will lead the way to a more customer-friendly E-government based on a services oriented (SOA) environment. This will require a new portal architecture based on the State’s enterprise architecture blueprint.

From a strategic perspective, the California Service Center will be the customer-facing site. It will intelligently redirect to the appropriate service centers which will be working together in a federated manner. This implies collaboration among the various departments to determine how to best organization the CSC (customer-facing) presence to best serve constituents. The ultimate goal is to make it as easy as possible to find things quickly and provide an efficient, electronic process that result in a “one-and-done” service for the user. This means service center delivery must be *friendly*, *trustworthy*, and *responsive*.

A federated portal environment strongly implies common branding and well implemented usability features. That is, key parts of pages look the same and key navigation features are standardized. The type of navigation structures must be agreed upon and implemented using the same model or template.

Collectively, the service centers will leverage shared services, such as Identity, Search, Real Simple Syndication (alerts, subscriptions, news), and Payment, as well the Employee HR features of the 21<sup>st</sup> Century/Payroll project that are made available via web services. It should be noted that large, packaged applications such as SAP or Oracle are not *shared services*. They are *shared applications*. To be SOA (and therefore, web services) compliant, the application functionality must be accessible via an XML interface – not by logging onto a portal or via a vendor proprietary application. So, this means that certain HR, Financials, Asset Management, etc. functions might be available via a web services interface as well as the native ERP application or its portal.

Shared services should be selected based on their business need and usefulness. Once selected, they should be built consistent with the state’s enterprise architecture, specifically SOA compliant and in line with state security standards for shared services. They also need to fit within the Technical Reference Framework and its Enterprise Architecture Model.

Operationally, service centers will utilize an enterprise infrastructure approach. That is, they will be deployed and managed in a small number of SOA-based data centers. Mission critical services will be deployed in a redundant manner across data centers to minimize single point of failures. This implies

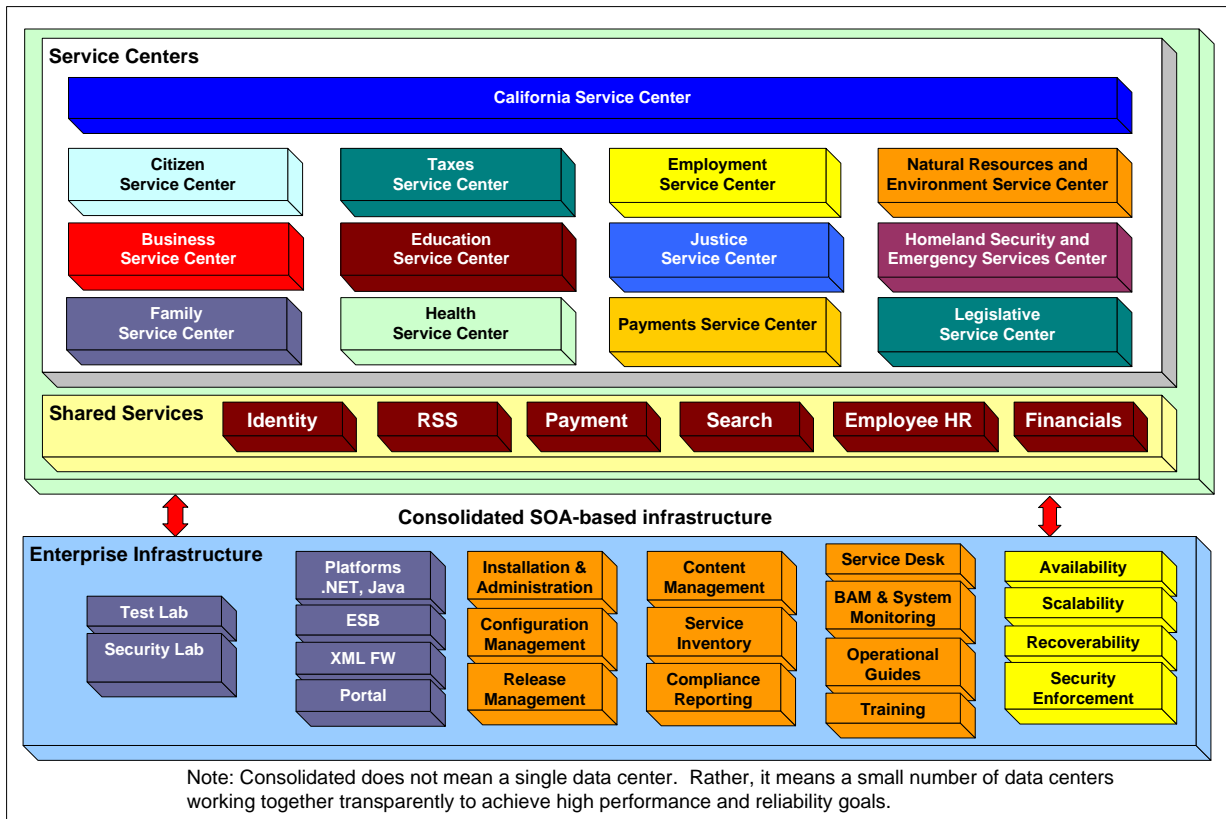
collaboration in the area of configuration and release management across data centers to achieve a high degree of availability, scalability, and reliability.

The CSC will be the primary entry point, however customers can still navigate through private industry search engines (such as Google, Yahoo, or MSN) as well as other government sites to reach CSC. So, this site must be designed for search engine friendliness.

# Consolidated Service Centers

## *Master and Customer-Oriented Service Centers*

One large service center for a state the size of California is not practical. A service center for each department is not manageable, cost effective, or user-centric. So, an approach where “initial interactions” come into CSC but then get federated to a relatively small number of intelligently grouped service centers seems to be the best answer. However, for this to be effective (and user friendly) some process and operational rules need to be established.



## *Consolidated Service Centers*

First, this pattern should be designed with the notion that its foremost priority is to provide a great user experience. This means layout of the initial page must be carefully considered. The things that the majority of users are interested in must be prominently displayed, or easy to locate. Search and navigation are also key areas that heavily influence usability. The content and data of the initial page must therefore be the result of collaboration among departments guided by statistics of how users are actually using (or attempting to use) the main site.

There must be agreement among service centers regarding how user interactions are handed off, processed, and returned. All service centers should have the same look and feel to the degree possible. This goes a long way towards achieving a positive user experience. Each specialty center should make it clear and easy for the user to return back to the master (California Service Center) site.

The CSC site will determine if user identity must be established. If so, it must also determine the type (citizen, business, employee, etc.) and invoke the appropriate identity authority which, if authenticated will provide a “token” (SOA security standards compliant message) containing the agreed upon user details. All service centers within a community of interest will trust this token and not re-authenticate the user. This is commonly known as WEB single sign on. Note, this ONLY established the user’s identity -- it does not deal with access privileges. This is the responsibility of each application. See [Identity Service](#) in this whitepaper for more details.

Many interoperability issues can be solved by selecting a good application integration platform, such as an enterprise service bus (ESB). ESB’s must be able to communicate transparently across service centers. Additionally, using standard web service interfaces can make interoperability an asset instead of a risk. However, the interfaces must adhere to proper identifiers and data definitions based on xml schemas. See [Service Center Infrastructure](#) in this whitepaper for more details.

The State CIO working in conjunction with department and agency directors will determine which service centers make sense based on business needs. Once determined, moving to these centers should be done in a consistent manner using common infrastructures, interfaces, and operational rules.

### ***Federated Development***

While it is important for all service centers to provide a standardized look and feel for a positive user experience, the content of each service center must be developed and maintained by department personnel who are the business experts. That is, while California will look like one big service center, it is actually a collection of loosely-coupled centers based on a common infrastructure.

Each of the conceptual service centers will provide business analysts and technical architects to ensure application accuracy and efficiency, as well as compliance with the state enterprise architecture. In order to achieve standardization across service center personnel, DTS in conjunction with DGS will provide a training program for service center developers (according to “California In-Touch”). This should include general training on how all the pieces fit within the state’s enterprise architecture vision as well as specific training on how to develop and use shared services. There should be business and technical “tracks” to this training. For example, tools need to be chosen to conduct training on business modeling, service modeling, and the details of translating these models into BPEL (which is the XML-based standard for orchestrating business processes in an SOA environment). Additionally, both business and technical developers will need training and guidance on best practices for determining the granularity and composition of web services. The technical developers will need Enterprise Service Bus training since this is a key component for XML messaging and service interoperability of the SOA infrastructure.

An enterprise-class tool needs to be chosen to serve as the enterprise repository for business models, processes, services, and portfolios. This tool would serve as the source for service inventory and usage, as well as a single place to go for application and project portfolio information. See [Enterprise Repository](#) in this whitepaper for more details.

Another key training component is how to effectively develop and manage shared services. This implies a process is needed to standardize the way requirements are gathered and determined that spans multiple departments. Once developed, training on how to use this process will be very important to minimize confusion and rework. See the [Training](#) section of this whitepaper for more details.

Following are a few examples of shared services. It is anticipated that many services will be developed to support federated service centers such as Address Verification, License Qualification, Educational Verification, etc.

# Shared Services

---

## Simply Shared Services

Initially, the California Enterprise Architecture Program laid the groundwork for shared services by defining Enterprise Services, and different levels of Shared Services. The primary differentiators between Enterprise and Shared services were scope and how the service was used. However, it is now thought that this probably added more confusion than necessary. So, going forward we will simply have Shared Services regardless of whether a service has “state” or “community of interest” scope or whether it is a “COTS” application of a web service that is consumed by a larger process.

A number of services should be built and managed with an “Enterprise mentality”. That is, they should have state-wide policies, be deployed across multiple data centers running redundant services, and backed by a single, unified help desk. This may require new service agreements among agencies and departments. In some cases, permanent working groups may be necessary to properly manage the service. Identity and Payment may be good examples of enterprise services that need an ongoing guidance body.

## Search Service

In [\*SOA Service Patterns White Paper\*](#) “*Enterprise Search Service and Federated Search Engines Patterns*”, it is recommended that an enterprise search functionality be based on a federated model. Accordingly, the California Service Center would provide the main state search service but redirect to other search engines as appropriate. Perhaps FTB would provide a really good search engine for taxes, Justice for police, courts, and legal information.

The search engines should expose their key services via Web Service Interfaces to ensure high interoperability. This allows search applications to choose among the interfaces, and provide a better user experience by incorporating personalization and profile features.

Developing a state-wide taxonomy would also be very beneficial and improve consistency and “expected results” across multiple search engines.

## RSS Service (alerts, subscriptions, and news)

Real Time Syndication (RSS) is an industry standard for publishing public information. Content is defined in XML files based on a standard schema definition. Any organization can provide a “feed” document which defines the content they wish to publish. The state RSS Service would be a centralized service that “polls” each of the feed sites, collects what has changed, indexes and stores the meta information in a common repository.

Clients (which could be anyone inside or outside of California), could use any of the many RSS News Readers, or use the one provided by CSC. The reader allows a user to “subscribe” to a category of information. They could then be automatically be notified by access choice (email, PDA, etc.) when information changes. Or, based on their preference, the information could simply remain in the repository until they invoke their reader.

By consolidating alerts, subscriptions, and news feeds into one enterprise service will dramatically improve the user experience, provide a consistent interface, and be much more efficient for the state to maintain.

See [SOA Service Patterns White Paper](#) “RSS Service Pattern” for more details.

## Identity Service

Many state applications require that a person be authenticated prior to allowing them access. Currently, this is usually handled by a login screen and back-end directory server in conjunction with application code. However, it is highly recommended that we move to a federated identity environment. The details of this concept are explained in the [SOA Security White Paper](#) document.

Developing an implementation of the federated identity model will require a collaborative effort among the “Identity Community of Interest” participants. However, this can start small and evolve into a true enterprise service. Each service center will need an Identity Service to handle user interactions that involve non-public information. One of the things that the collaborative group needs to determine is how to identify interactions that require the authentication of a person’s identity. This may be accomplished by a standardized encoding scheme of the URL (only for SSL connections), or by hidden fields during a form submission. In the future, more sophisticated methods such as personal identity cards, or RFID-based cards might be used to extract identity information and pass it to the Identity Service.

Because there are many complex issues with a federated identity service and processes need to be developed to properly manage identity elements on an on-going basis, an SOA Security Working Group will be formed. This group will be responsible for establishing Identity and its security-related processes, define security levels and which SOA security standards will be used for each level, as well as exactly what data will be included in the “token” and how that data will be protected.

While this should be a permanent guidance group, its membership should change based on the particular identity elements being worked on at that point in time.

## Payment Service

Payment would be a great shared service that could be used by any government entity that collects payments from customers. This service should provide and maintain all the necessary interfaces to the back-end payment service providers so departments would not have to concern themselves with changes to state-wide contracts with those providers.

The Federal government has set a good example of elevating a shared payment service to a Service Center status. Please look at the [pay.gov](http://pay.gov) site.

From the SOA perspective, the most important aspect of a payment shared service is its interface. This is what consumers of the service will see and use. To be SOA compliant, the interface must be document-based. That is, not RPC based. But more importantly, the interface must be broad enough to accommodate the payment community of interest. Because it is anticipated that this service will have many users, it must be a very stable and robust interface. Also, considerable thought must be given to hardware and network architectures since availability and scalability will be critical success factors of this service.

The payment service interface should specify the identifier, message formats, and the protocol. A properly defined service interface will ensure the loose coupling of this component across applications. Identifier is the name and "address" of this service specified as a URL. Formats are the XML message structures that will be used by this service. The protocol defines the rules for interaction between the consumer and provider. For example, the data input and output structures that the service will require must be defined as "tags" within the input and output messages. A key component of web services is XML "extensibility". That is, the interface should be designed so it can be easily extended WITHOUT breaking existing consumers and provider contracts.

Security is another aspect to be considered. Will the payment service allow customers to record their payment information (debit/credit card number, etc.) for the purpose of reusing that information on multiple transactions over time, or to facilitate automatic payments? If so, as custodian of this information the payment service will need strict rules for protecting this data as well as the capability to keep the data current.

It is assumed that the payment service will offer different levels of logging. All transactions could be logged or just certain types of transactions. The amount of detail that is logged for each transaction could also be configurable. In both of the above cases, logging configuration should be runtime configurable and not part of the code so it can be easily changed. "Pre-canned" usage and auditing reports could be developed to assist Auditors and managers. For example, a report might show payment service usage by interface type. Another report might provide "drill down" capability for the purpose of auditing. Of course the results of these auditing reports would depend on the level of logging.

Finally, the payment service must provide the following capabilities to the consuming applications that will utilize this service:

- The appropriate payment transaction information to update back-end department accounting systems.
- Interfaces to allow Accounting and Customer Service staff to query payment transactions.

## **Employee HR and Financial Services**

Certain services can be shared across service centers (a few examples are Identity, RSS, Search, Payment, and Address Verification). However, there is another class of shared applications usually associated with ERP or COTS types of applications

Most "COTS" type of applications are not shared *services* – they are either stand-alone applications or they may be shared *applications*. If a *web service interface* were provided to assets stored in SAP Financials, then this service could be invoked and the resulting data incorporated into an application of choice. This is quite different from querying asset information via the SAP Financial GUI screens. Another example, we might choose to make certain geospatial data available via a web service interface. It could then be invoked and the resulting data could be used in different applications. Again, this is different than using a Google Earth or ESRI client application.

SCO's 21<sup>st</sup> Century Project has been underway for some time with specific deliverables in the FY2007/2008 timeframe. The main focus of this effort is to standardize the state's employee HR functions. This will primarily be a shared *application* with user access via the SAP portal. Since employee identification will be a component of this system, it is suggested that SCO package this functionality to become the Identity Authority for employees. Then, any service center that is handling related employee interactions could use the SCO Identity Authority service to authenticate employees. Once authenticated, other applications within the community of interest would "trust" this service and not re-authenticate the employee. Additional employee information that currently would only be

available via the SAP Portal might be exposed via web service interfaces. This functionality would then be shared services that could be invoked (consumed) by a variety of applications.

### **Other Shared Services**

Over time, business requirements will identify additional needs for shared services. Some examples might be Address Verification, Criminal Background Check, Academic Credentials Verification, Real Estate Risk (Flood, Earthquake, Toxic Chemical, etc.). These services will be developed in a [federated](#) manner, deployed and registered with an SOA-based [consolidated data center](#).

## Consolidated Infrastructure

---

The enterprise infrastructure will support the California Service Center, federated service centers, as well as many other non-service center customers. Therefore, it must have a high degree of accessibility, scalability, and performance in terms of fast user responses. This section details some of the key components that will play a key part in ensuring the above goals are met.

### ***Enterprise Repository***

A single, enterprise repository tool is needed to gather information about enterprise architecture models, department applications, shared service inventory and usage, as well as pointers to shared service code packages. Therefore this repository should be purchased and implemented as a shared enterprise tool with the capability to establish different types of users, and appropriate levels of security.

This repository should be configured to hold details of the Business Reference Models (BRM), Service Reference Models (SRM), Data Reference Models (DRM), and Technical Reference Models (TRM). For example, the Technical Reference Framework templates would be stored in this repository. As departments fill out the templates they could also be stored in the department section.

The repository tool should have a good search screen. Pre-canned reports should also be built. Some examples are ones that show shared service usage sorted by service name. Another one could show all shared services used by a department, and third one that shows how shared services in the SRM are related to business services in the BRM.

Since there will be many users of this tool, maintenance procedures will be important. This includes repository setup and on-going modification, user access management, and appropriate backup and recovery. It must also have high availability.

One last, but important feature of the enterprise repository is it would be a great place to store “canned” requirements language for RFI/RFPs.

CollabNet, Proforma, Troux, plus a number of other companies provide products that could potentially serve as the base tool for the enterprise repository.

### ***Platforms***

Web services are developed in either .NET or J2EE environments. In the case of .NET most developers use Microsoft’s Visual Studio (or Visual Web Developer). In the J2EE world, most vendor products are based on Eclipse ([www.eclipse.org](http://www.eclipse.org)). Both of these tools have built-in resources for packaging services and deploying them into their respective runtime environments. Data centers supporting SOA services must support both environments since customer departments develop on both platforms.

The mainframe can also be a platform particularly as part of a migration strategy via placing web service interfaces on mainframe resources.

### ***SOA Infrastructure***

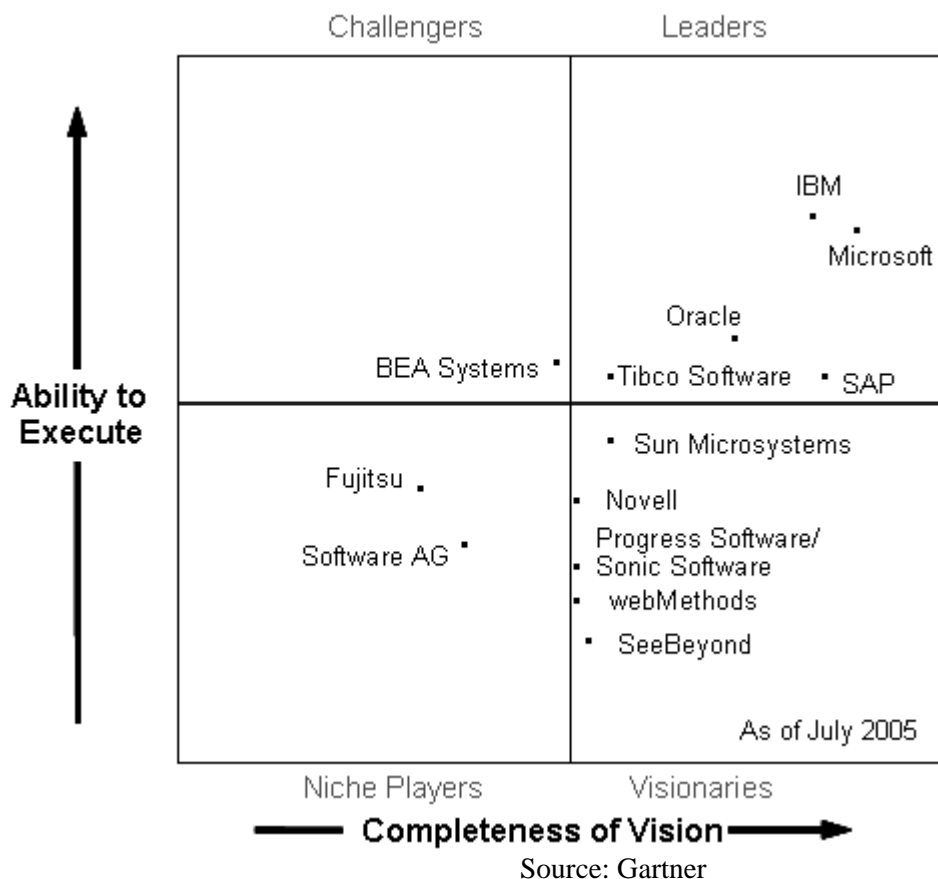
From an SOA perspective, an infrastructure must support XML messaging. This includes message routing, mediation, and xml transformation. Consumers and providers of a web service always interact via XML messages (RPC-based SOAP messages no longer meet SOA requirements). There are

different types of messages (for example, SOAP and WSDL) which conform to their respective schemas.

*Application integration and service interoperability* are of paramount importance in a shared services environment. Because applications are located on many different hardware and software platforms, connecting these environments in a manner that meets availability, scalability, and user performance expectations, uniform service interoperability platforms must be carefully chosen. These typically fall into two categories: For Windows-based platforms, Microsoft handles standards-based messaging and application connectivity via incorporating Windows Communication Framework functionality into an upcoming release called Windows Vista. In the Java world, there are quite a few choices. Some of the more popular ones are IBM WebSphere ESB, BEA AquaLogic Service Bus, Oracle Fusion, Cape Clear ESB, Sonic Software ESB, SAP NetWeaver, and others.

According to Gartner's "Magic Quadrant", the SOA Infrastructure leaders are listed below.

[http://mediaproducts.gartner.com/gc/webletter/microsoft4\\_enterprise/2005/article15/article15.html](http://mediaproducts.gartner.com/gc/webletter/microsoft4_enterprise/2005/article15/article15.html)



So an SOA infrastructure must have support for XML message processing and application integration. This functionality is normally handled by an ESB. However, there will be more than one ESB so it is important that the ESB chosen meet SOA industry standards particularly in the area of XML messaging, application APIs (adaptors/connectors) as well as BPEL support. For availability and scalability reasons, web services will run on multiple servers so ESB's must communicate based on standard protocols and message formats.

See [Security](#) section of this document for additional infrastructure requirements.

## ***Labs***

### **Test & QA Lab**

Data centers hosting shared services will need a test and QA lab where new or modified services can be thoroughly (regression) tested. Those that pass will need to be assigned to a specific profile release (similar to WS-I profiles <http://www.ws-i.org/>). This is a way of tracking which versions of which services are known to work together. Whenever, the revision is rolled on a service, it must be re-certified to the profile. Those services that fail these tests need to go back to the developers for resolution. The Test & QA lab needs to mirror the production environment based on the type of testing. For example, if the tests are to measure performance and scalability, then the size hardware in the test environment must match the size of the hardware in the production environment. Otherwise, the test lab hardware can usually be scaled down if only testing functionality and interoperability.

This lab could also be used to test interoperability between data centers (to ensure appropriate availability). Additionally, it could serve as the test area for recoverability testing that is usually associated with a disaster recovery or other type of readiness plan.

### **Security Lab**

It might also be useful to have a separate security lab where “security proof-of-concepts” could be conducted. This would be useful when developing state standards for different levels of security elements as required by certain web services that are shared.

## ***Technical Administration***

### **Installation**

Initially, as the SOA infrastructure is built out most of the administration work will focus on installing the application server platform components and the ESB. This includes installing and configuring the Enterprise Repository. If a UDDI is defined, this would also require installation and setup. However, once the basic infrastructure components are in place and configured, the majority of on-going administration will focus on managing shared services and keeping the enterprise repository up to date.

The above implies in depth skills in Microsoft .NET and J2EE components. It will also require vendor specific product skills (for example IBM WebSphere, Cape Clear ESB, Oracle Fusion, etc.).

### **Troubleshooting**

When problems arise with shared services, new troubleshooting skills will be required to pinpoint the problem. In addition to normal network or machine access issues, problems could arise with the web services infrastructure such as an ESB, .NET Framework, J2EE application server, or web application server components. The troubleshooting process can be further complicated if the problem is centered on a service composition (a service composed of other services) or BPEL (orchestrated services) process.

The user’s description of the problem is always the starting point, but often this is not very useful in locating the underlying problem. Log analysis is always a good next step, however in a web services environment there are usually multiple logs to review. The real cause of the problem is often indicated in either Java or .NET error messages. So, technical administration personnel will need additional skills

to interpret these messages in order to effectively solve service problems. It is likely that a number of problems will be the result of improperly configured JVMs (Java Virtual Machine). For example, stack overflows, out of resources, increased queuing for object services, or database waits. This usually occurs because either proper performance testing wasn't done during QA testing, or because a substantial number of new users have come on line for a shared service.

Obviously, procedures need to be in place to proactively monitor shared services and resource demand increases. Automated tools can help identify potential problems in advance. Tools like Mercury Roadrunner can pinpoint performance problems during QA testing. System monitoring and BAM tools can help during production runtime.

If a significant problem occurs, a process to either bring more resources on line or re-route requests to the same services in a different data center must be put into place --- and tested.

## **Backup & Version Rollback**

Backup copies of all components that make up a shared service must be taken every time the service is versioned (or a patch release is applied). Obviously, the backup copies should be located separately from the data center where they are deployed.

Operational procedures should be in place so there is no guesswork (or forgotten steps) in the backup. We do not want to be in a recovery situation only to find out a key file was missing in the backup. See [Disaster Recovery](#) for more details.

## **Service Management**

The following categories of operational management have ITIL processes in place and some of these processes will need to be modified to handle SOA service specific issues.

## **Configuration & Release Management**

A process needs to be developed for determining how to configure the .NET and J2EE environments in support of shared services in the operational data centers. Developers need to know this process so new (or modified) services can be submitted in a standard way. This process needs to include a standard mechanism to handle feedback to developers when services don't pass the [compliance and testing process](#) for new or modified services. Developers also need to be aware of the testing process details (including [performance and availability](#)) so they can catch problems during their unit testing.

Release management includes deploying the EAR file (J2EE) or .NET package onto production servers. Versioning the service, updating the inventory and usage section of the Enterprise Repository are also part of this function. Additionally, if the service is to be deployed in additional service centers, a process must be put into place to coordinate releases.

## **Service Compliance Testing**

As new services are developed they will be unit tested then given to data centers for compliance testing. Note if this is a service that will be deployed to multiple data centers, then integration, QA testing, and configuration & release management must be synchronized across the data centers. Otherwise, availability and scalability contracts may be comprised.

Data center compliance testing is not simply repeating the unit tests by developers. Its main focus is to determine if the shared service will play nicely with other shared services on a

common platform, as well as determine if it will meet performance and availability requirements. So, situations will occur where a service passes unit testing but not compliance testing due to differences in the way machines are configured in the data center verse in a particular development lab. Remember, there will be many development labs both inside and outside the state. So, it is a given that they will not be configured exactly the same. This must be reconciled during compliance testing.

In reality, some service releases will have a higher priority than others. So, an escalation procedure is needed to ensure the right resources are put into place. This likely includes consulting resources as well as state employees.

## **Service Performance**

Service performance metrics are generally focused on (but not limited to) availability and scalability. How well a shared service performs is often the result of data center deployment decisions. While they can be the result of an incorrect design, more often than not they are the result of incorrect hardware (or network) sizing, runtime application server configuration, availability configuration across data centers.

This brings up the issue of how and where large scale performance, scalability, and availability testing be done. It is recommended in the [Labs](#) section of this document that it is done in the Test and QA Lab. However, this can get complicated since this lab is for approving (or rejecting) all shared services. It is conceivable that some availability or scalability tests could be quite involved and require days or weeks to fully test. In some situations, other service testing may be put on hold until this has completed. They may or may not be acceptable. So, outside labs provided by the development organization (or its consultants) may be required for some large scale tests. This must be factored into the compliance process requirements, project timelines and budgets.

In some cases, standard Service Level Agreements may be required. This may be complicated by the fact that this is a shared service so the “users list” will probably change over time. So, shared service SLA’s need to be documented where performance is based on a changing set (and quite possibly increasing) set of users.

## **Common Service Desk**

Problems with shared services, and particularly composite and orchestrated services, should be managed via a common service desk. This includes cross data center processes in cases where a service is running in multiple data centers. The user should only have to open one ticket which is managed by one organization regardless of who (or where) the actual troubleshooting and fix work is done. Obviously, this can be a complicated process but it must be thought through in advance with appropriate procedures put into place. The last thing we want is for the Payment Service to go down and people “start pointing fingers”. Not a good way to increase adoption of shared services!

All data centers have current help desk policies. However, they need to be modified to handle the additional constraints imposed by the shared service environment.

## **Service Recoverability & Disaster Management**

Shared services must be part of the overall data center disaster management and business interruption plans. Policies of how to handle cross data center issues as well as version synchronization issues may be stated here. The risks of a shared service outage must be identified and managed. Of course the degree of risk is probably directly related to the scope of the shared service. That is, a service that is

consumed state-wide probably carries greater risk than one that is used by only a few departments. However, it is certainly conceivable that even though a service has a small usage base, it might be considered a mission critical service. So, risk management plans need to factor in the scope issue.

A key part of a disaster recovery plan is how to get the service back on line. In many cases, this means restoring service components from a backup. So, it is critical that backups are up to date. It is equally important that the backups are accessible and qualified technical administrators are available to install the backups. Among other things, this implies an adequate personnel training program to test the procedures.

Particular attention must be given to shared services that are running in multiple data centers. In this situation, it may not just be a simple matter of rolling back a version because we would have a version mismatch across centers which might cause problems. Ideally, this should not be a problem if there has not been any change to the service interface. Generally, the implementation details of a service can change as long as the interface is consistent. However, if the business logic or business rules have changed, then this could be a problem. The service will work, but probably not produce the expected result.

## Training

A good training program could dramatically speed up SOA adoption as well as help achieve standardization across departments.



Training should not be limited to developers. Since these are shared services, all roles need to understand how all the pieces fit together from an enterprise perspective. So, one way to achieve this is to offer introductory training that covers all the SOA topics in an abbreviated format. In other words, this is SOA – its parts and pieces. Participants in all roles (Manager, Developer, Business Analyst, and Technical Admin) would receive this training first. It would be the foundation (or prerequisite) training.

Then, it is anticipated that there would be business and technical “tracks” (or “versions”) to more detailed training segments. For example, the business track of Web Services Security could address the business issues (policies, types of data, privacy issues, security levels, etc.) and the technical track of Web Services Security could handle the various options for implementing the security policies including protocols, message definitions, XML firewall and ESB security configuration.

Another example might be Shared, Composite, and Orchestrated Services. The business track would focus on determining how services should be shared while the technical track would model the service and create BPEL documents.

### ***Business Activity Monitoring (BAM)***

Business activity monitoring (BAM) sprang up in 2001 and it provides a dashboard readout of key performance indicators updated in real time. Some say BAM is a function, not an application. BAM is not business intelligence, which is rearview information, not real-time. BI users tend to be business analysts; BAM users are typically line-of-business managers. BAM information tends to be focused rather than broad.

BAM enables organizations to leverage business analytics to gain a real-time insight into daily business operations. This analytical insight helps business users quickly identify operational inefficiencies and predict potential business problems. BAM integrates business intelligence (BI) with business transaction processing.

"In a data warehouse, you want to collect lots of information that you can slice and dice a hundred ways for future analysis," says David Kelly, president of Upside Research. "BAM alerts the user based on a few carefully chosen business metrics and thresholds that affect the bottom line directly and require quick action."

Debbie Rosen, executive vice president of worldwide marketing at webMethods, agrees. "It's like the difference between looking at a live stock ticker symbol versus looking at the stock's 52 week history and company financial information. You want both."

Whereas BI pulls information from a data warehouse, BAM tends to tap information sources directly, linking to a message broker for example, or using a very low-latency data store. Information gleaned from BI, however, often plays a role in choosing hot spots for BAM to monitor. "Your BI analyst will tell you these are the predictive measures and types of things that lead to problems, so you can start using BAM to catch those things as they happen," says Scott Fingerhut, senior product marketing manager at Tibco. In fact, some BAM solutions continuously compare BAM information to historical data to provide context.

Just about everyone agrees that a successful BAM implementation comes from starting small, using what you have, modeling relevant business processes, focusing tightly on a few important Key Performance Indicators, and marrying business users with IT.



Oracle BAM screen

Integration is a significant part of the BAM implementation process, so if you've already gone through EAI or some other form of business integration, it makes sense to leverage what you have in place and seriously consider what BAM capabilities your integration vendor offers. BAM dashboards require close collaboration of business and IT. The business-process owner is the one who knows exactly where in the process she lacks visibility and where things fail on a bad day.

The BAM solution should provide users with easy tools, such as drop-down menus to manipulate KPIs and thresholds as business conditions change, so users don't have to go back to IT each time they want to make a small adjustment.

The objective of a BAM project is to *monitor*, *analyze* and *report* on the performance of business operations, and for business users to act on this performance information to improve business operations. Let's examine each of these steps in turn.

*Monitoring* involves tracking and collecting information about a business operation. Monitoring is easier if the operation being tracked has been implemented in terms of the business activities required to carry out the operation. This granularity enables the monitoring task to track just those activities that are important from a performance perspective. Without this granularity, the monitoring task can only track the complete business operation. A business process and its underlying activities can be

implemented using business process management software and an underlying service-oriented architecture (SOA) that defines each business activity as a separate callable service.

The *analysis* step of BAM processes the information collected by the monitoring step and creates a set of metrics documenting the performance of the monitored business activities. This analysis can be done synchronously and in-line as a part of the main business process, as in the procurement example previously mentioned, or it can be done outside of the main business process. In this latter case, the monitoring information can be routed to an asynchronous *in-line* process for analysis, or it may be stored in a message queue or persistent store for *off-line* analysis by an independent application. The method chosen will depend on how quickly the analysis has to be done and acted upon (i.e., on the timing requirements of the business application).

In-line analysis of monitoring information is supported by several business process management and application integration vendors such as IBM and TIBCO. It is often these vendors that still use BAM terminology. Off-line processing of monitoring information is supported by independent BAM vendors, such as Celequest, and a number of BI vendors. Here, vendors use terms such as operational business performance management and operational dashboard builder, rather than BAM. Operational business performance management and BAM are conceptually the same.

The *reporting* step of BAM involves displaying or delivering the results of the analysis step. This may be done via a portal, dashboard, e-mail, instant message, etc., depending on the preference of the user and how quickly the information must be acted upon. In some situations, a rules-driven alert may also be sent to highlight important information.

The main objective of BAM is to be able to monitor and analyze business operations and deliver information to business users so that they can react rapidly to business requirements and problems. Most of the decisions and actions taken using BAM are reactive in nature - the user acts after a business situation has occurred. Using predictive BI techniques in conjunction with BAM, however, makes it possible to detect patterns in business operations and predict business issues before they occur. An example here would be assessing the risk of giving someone a license or determining that an insurance claim is fraudulent and should be handled manually.

The *decision and action-taking* step that follows the analysis and delivery of information from a BAM application is the most critical aspect of a BAM project. It is this step that enables business users to optimize business operations and align those operations with the goals of the department. This is why the output from a BAM application must be tied to a business goal. This goal may be a specific target (reducing call center action time, for example) generated by a balanced scorecard, planning or budgeting application. It may also be a more complex goal such as determining the actual cost of each business activity in a business process to provide a complete picture of the costs of doing business. In this case, the results can be used in conjunction with BI software such as activity-based management.

It is important to realize that BAM is just a piece of the technology puzzle required to support the management and optimization of daily business operations. Other key pieces of the puzzle include business process management (including a service-oriented architecture) and business intelligence techniques.

From a data center perspective, BAM tools could be used to monitor service levels. This would provide a real-time view of shared services performance.

Example BAM vendors:

IBM WebSphere Business Integration Monitor  
Microsoft BizTalk Server  
Tibco – OopsFactor  
Oracle Business Activity Monitor  
SAP  
iSphere  
firstRain  
Celequest  
Informatica  
Vitria  
Micromuse  
Computer Associates  
BMC

## **Security**

See [SOA Security White Paper](#) for a detailed description of message-based security. The California Service Centers will need to support the security-related components and configurations as determined by project design teams. This includes protocols, XML message and data formats, as well as XML firewalls. In particular, the firewalls will need to “look” inside a message and determine whether or not to pass the message or return it.

Web services security is based on many standards which continue to evolve. It will be important to stay on top of these standards as they merge or new ones appear. It is the data center’s responsibility to enforce the security mechanisms defined by project designers and implemented in deployment packages. It is likely that some of the security components will require collaboration across shared services and across organizations and data centers.

## Packaged Portals

“A few years ago the term "portal" emerged in connection with major websites such as AltaVista, Google, MSN and Yahoo!. A portal offered a single entry-point to content and functionality. In today's world we have portal software, which is roughly an attempt to recreate Yahoo! within the enterprise.”

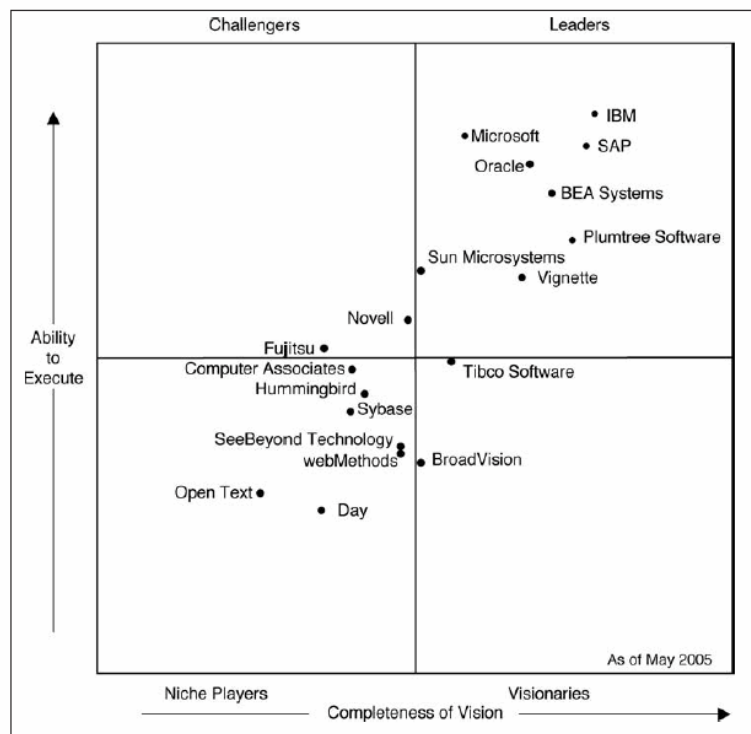
*Janus Boye, CMS Watch*

A recent AMR Research survey revealed that 72 percent of companies are in the process of reevaluating their current supplier portal vendors. Supplier portals are the software-based communications engines that are designed as a means to enable companies to better integrate information, foster knowledge sharing, improve productivity, and support supplier collaboration with their suppliers.

“Improved supplier portal offerings from ERP vendors are enabling organizations to think more strategically about their supplier portal investments, architectures, and choice of vendors. ERP vendors are now repackaging their supplier-facing functionality into cohesive portal products that provide customers with incremental short-term gains and support more complex business processes for long-term success. Many companies are now choosing to replace current portal vendors with the supply portal offerings of their existing Enterprise Resource Planning (ERP) vendors.” *AMR Research*

“The portal product market is growing, but the number of vendors in the market is shrinking. Portal products are purchased as point products, but suites now garner a growing amount of portal product sales. The Horizontal Portal Product Magic Quadrant shows interesting changes in vendor positions. *Gartner, Inc. 2005*

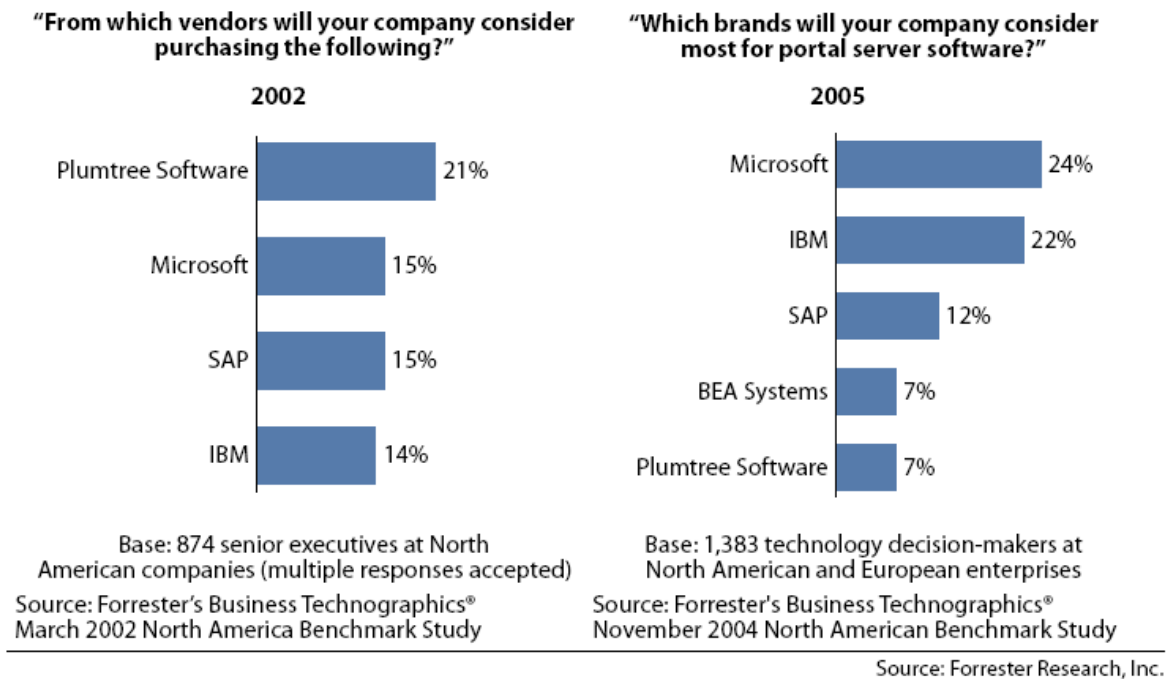
**Figure 1. Magic Quadrant for Horizontal Portal Products, 2005**



Source: Gartner (May 2005)

“Vendors like Plumtree Software (acquired by BEA in 2005) and Epicentric created the portal server market in the late 1990’s by offering services with technical features that just didn’t exist within firms’ existing IT infrastructure. Now those features – like UI abstraction, integration, workflow, and delegated administration – have been co-opted, improved, and embedded in general-purpose infrastructure platforms from vendors like IBM, BEA Systems, Oracle, and Microsoft. The standalone portal server market is gone, absorbed into infrastructure vendors’ app server platforms and emerging interaction platforms.” *Forrester Research, Inc. 2005, Nate L. Root*

**Figure 3** The Portal Vendor Popularity Contest, 2002 To 2005

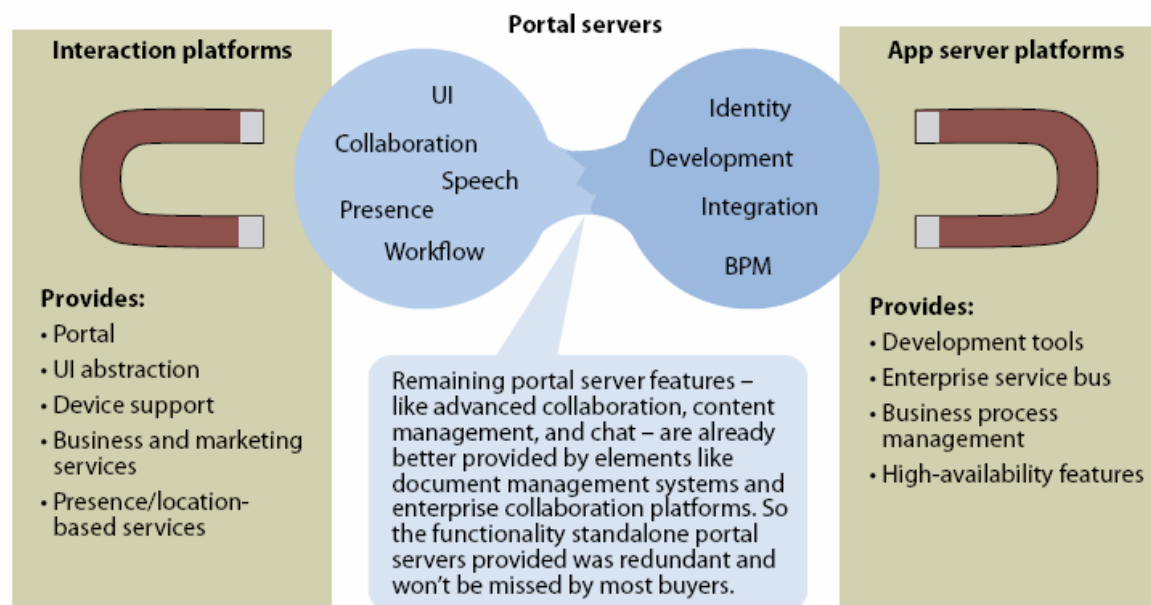


Enterprise portals rank in the top 10 of CIO technology focus areas in many surveys. Horizontal portal products continue to be in the engine that drives enterprise portals. Horizontal portal products can be packaged in suites (for example, in application platform suites or smart enterprise suites) or can be acquired stand-alone.

The largely independent software vendors (ISV) in the horizontal portal product market continue to take market share from midsize and smaller players. Large ISV’s have always run the “stack play,” but with the emergence of the business process platform this play takes on a new meaning. The trend in movement toward large ISVs will continue to take share from midsize and small vendors, forcing them to pursue a niche strategy or to seek acquisition. *Gartner, Inc. 2005*

“The ideal platform for building a portal isn’t a portal server anymore, but it’s not any other single piece of IT infrastructure either. Tomorrow’s portal sites will be built using services from two main architectural elements as traditional portal server features are divided up and absorbed (see Figure 4). *Forrester Research, Inc. 2005, Nate L. Root*

**Figure 4** Portal Servers Are Absorbed Into Interaction Platforms And App Server Platforms



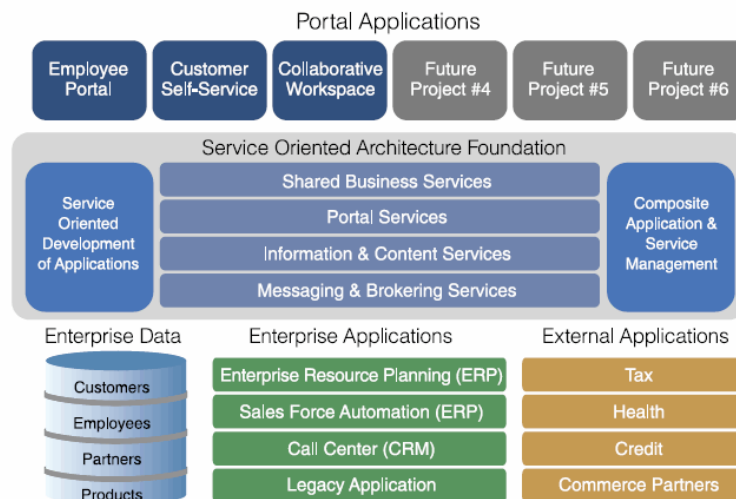
Source: Forrester Research, Inc.

This quote from SAP's Chairman clearly states the importance of portals to the platform providers:  
 "Enterprise portals are the key to corporate success and the linchpin of SAP's future strategy" *SAP AG Chief Executive Officer (CEO) and Co-Chairman Hasso Plattner*

IBM, BEA, and Oracle all tie their portal strategies to their SOA offerings:  
 "BEA WebLogic Portal simplifies the production and management of custom-fit portals, allowing you to leverage a shared services environment to roll out changes with minimal complexity and effort." -- *BEA Systems, Inc.*

"WebSphere Portal: An on-ramp to a service oriented architecture" -- *IBM October 2005.*

#### IBM WebSphere Portal and SOA



And Gartner agrees:

“As organizations search for a way to leverage a service oriented architecture, many can use portal products as a first step.” *Gartner, Inc. “Portals Provide a Fast Track to SOA” G. Phifer, July 2005*

An interesting perspective on government portals:

“Most visitors of local government sites don’t want to rearrange their own pages and portlets. They just want to know when the next garbage collection will be or what times the local swimming pool opens. If my local council proposed offering me a portal, I would tell them to spend the money on a decent search engine. Forget about the personalized portal experience – just show me the right content and show it to me quickly”. *Janus Boye, CMS Watch*

In the Java community, some standards have emerged around portals. JSR 168 and WSRP (Web Services for Remote Portlets), two portlet standards developed by the Java Community Process and OASIS standards bodies, respectively, aim to let portal components be deployed across a variety of platforms. JSR 168 was approved in early October, while WSRP was finalized in mid-September. So, the above standards should help reduce the interoperability risks of using multi-vendor portlets.